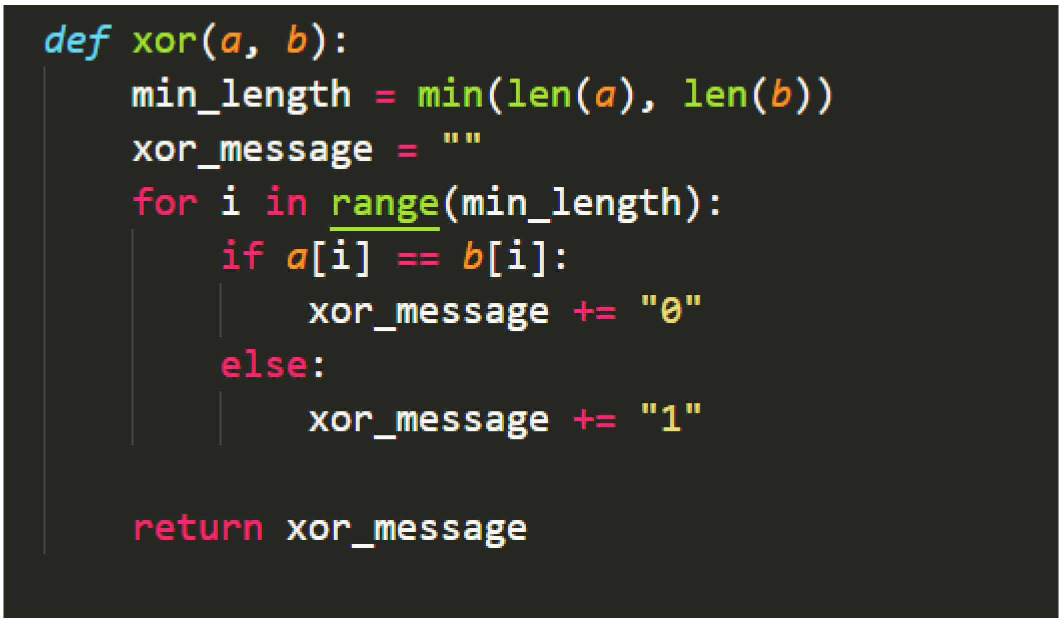
**Task 4 MAC**

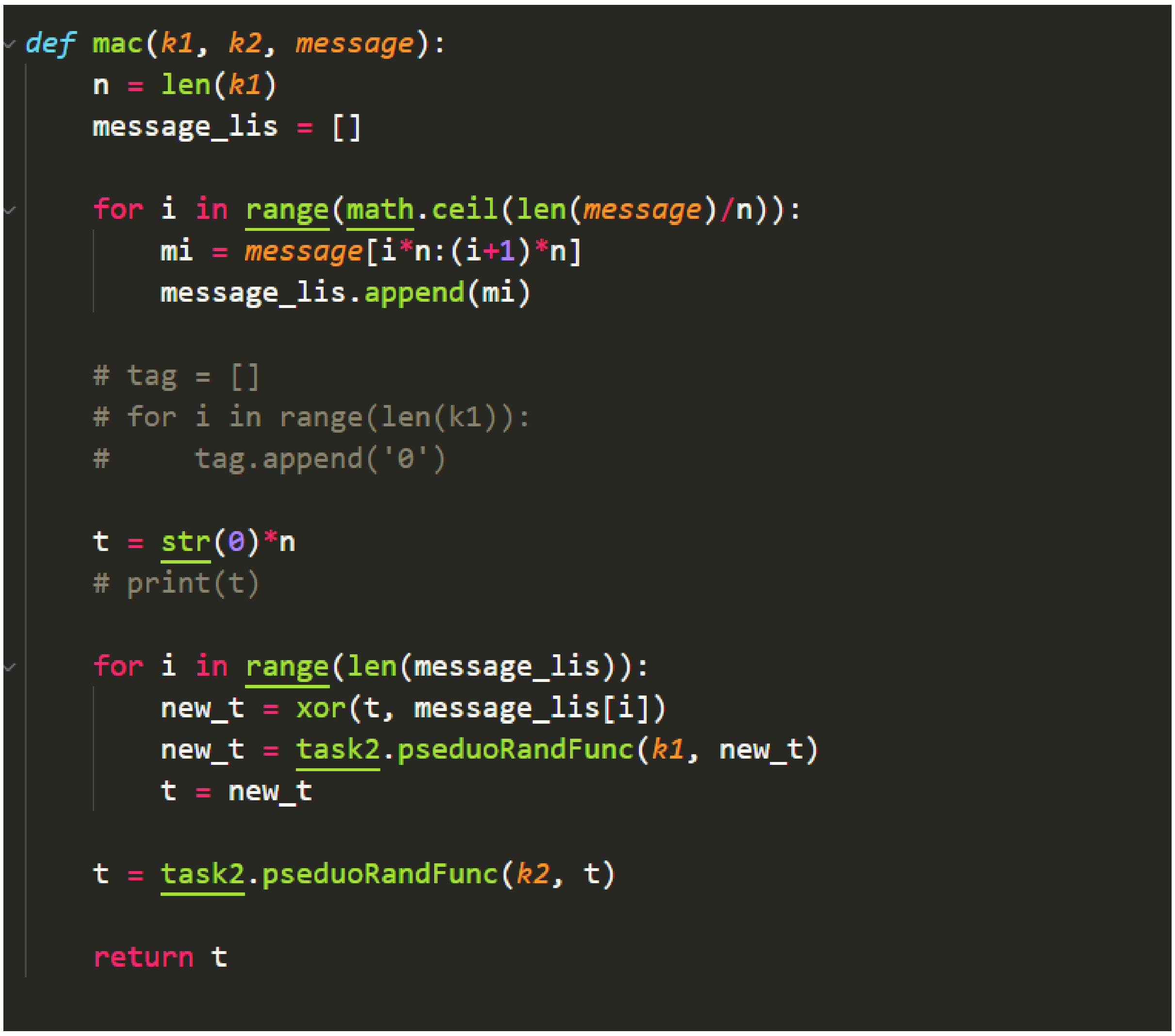


```python
def xor(a, b):
    min_length = min(len(a), len(b))
    xor_message = ""
    for i in range(min_length):
        if a[i] == b[i]:
            xor_message += "0"
        else:
            xor_message += "1"

    return xor_message
```

**XOR()**

Takes two bit strings as input and returns their XOR value as output.


```python
def mac(k1, k2, message):
    n = len(k1)
    message_lis = []

    for i in range(math.ceil(len(message)/n)):
        mi = message[i*n:(i+1)*n]
        message_lis.append(mi)

    # tag = []
    # for i in range(len(k1)):
    #     tag.append('0')

    t = str(0)*n
    # print(t)

    for i in range(len(message_lis)):
        new_t = xor(t, message_lis[i])
        new_t = task2.pseduoRandFunc(k1, new_t)
        t = new_t

    t = task2.pseduoRandFunc(k2, t)

    return t
```

**Mac()**

Driver and main code of the task. Follows 2 keys secure mac procedure, takes Key1, Key2, and message from user as input, distributes the message bits into blocks of equal length. Calculates mac of each block and appends the result. Calls prf on the final mac with second key and returns the final output.

**Sample Input**

Key1: 10011

Key2: 11000

Message: 1010101010100001111000111

**Sample Output**

MAC: 10110